Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1997 Proceedings

Americas Conference on Information Systems (AMCIS)

8-15-1997

On the Theoretical Foundation forMultidatabase Query Graph

J. Leon Zhao
zhao@uxmail.ust.hk

Jun Yuan
yuanjun@uxmail.ust.hk

On the Theoretical Foundation for Multidatabase Query Graph

J. Leon Zhao and Jun Yuan
Dept. of Info. & Syst. Mgmt., HKUST, Clear Water Bay, Kowloon, Hong Kong
Email: {zhao, yuanjun}@uxmail.ust.hk

1. Motivation

A new challenge to database researchers in the Internet era is to develop a more user-friendly query interface that is easy to use and powerful enough for formulating complex queries towards multiple databases [2,10]. We propose a multidatabase query graph (MQG) technique to meet these two requirements. The MQG technique uses concept hierarchies [1,7] as a graphical query interface [3,11] in the heterogeneous database environment, extending the schema coordination approach [12,13].

2. Multidatabase Query Graph (MQG)

Assume that the relational model is given for each export database, and therefore, the names of the entities and relationships, the functional dependencies between the attributes, and the semantics of the entities and attributes are known.

Definition: Link. We refer to a two-way join between a key and a foreign key as a link.

Definition: Path. If there is a series of links between two objects, we say that there is a path between them. We assign a semantic meaning to each path between two objects and written as $_{ij}$. A path is then written as $p<O_i, O_j, _{ij}>$. Note that O_i and O_j are the object names, and $_{ij}$ specifies the path and its semantic meaning as $\{l_{i,x1}\ l_{x1,x2}\ ...l_{xn,j};\ _{ij}\}$, where x_i, $i[1, 2, ..., n]$ identifies an object on the path, $l_{i,k}$ is a specification of a link, written as $< O_i, O_k, a, b>$, and a, b are key and foreign key attributes from O_i, O_k. The symbol is a concatenation operator that connects two links on the path. Since there can be more than one path between two objects, the set of paths between two objects is written as $P<O_i, O_j, _{ij}>$, where $_{ij} = \{^{k}_{ij}, k[1, 2, ..., n_{ij}]\}$, where $^{k}_{ij}$ is the k^{th} path, and n_{ij} is the number of paths between O_i and O_j.

Definition: Association. The semantic meaning of a path between two objects.

Definition: Object Hierarchies. Hierarchies of objects resulting from generalization/specialization relationships among objects. The object hierarchies may be explicitly defined by ISA relationships in an ER diagram or exist implicitly in the relations of a database.

Definition: Semantic Query Graph. A semantic query graph is written as $^{S}(^{S}, ^{S}, ^{S}, ^{S})$, where ^{S} is the set of objects, ^{S} is the set of object hierarchies, ^{S} is the set of attributes, and ^{S} is the set of paths between objects.

Definition: Multidatabase Query Graph (MQG). A multidatabase query graph maps to a set of SQGs corresponding to the multiple databases and is written as $^{M}(^{M}, ^{M}, ^{M}, ^{M}, ^{M})$, where ^{M} is the set of objects, ^{M} is the set of object hierarchies, ^{M} is the set of attributes, ^{M} is the set of paths between objects, and ^{M} is the set of database names.

3. Derivation of MQG

Definition: Export Schema. We assume that each export database D_i provides an export schema in the form of ES_i(rel.attr, semantic, scale, type, pointer), i [1, 2, ..., n], where rel.attr is the concatenation of

relation and attribute, semantic is the semantic meaning of the attribute, scale is the unit of the attribute if it is numeric or the meaning of values if it is symbolic, type can be either key, nonkey, or foreign key, and pointer contains the attributes to which the given attribute points. A foreign key can be a recursive foreign key specifying a recursive join, or an ISA foreign key specifying a generalization/specialization relationship. All keys may be composite keys.

Definition: Rules of pointer assignment. The rules of pointer assignment are: (1) A nonkey attribute points to its key attribute(s); (2) A key attribute and its foreign keys point to each other; (3) A foreign key and its home key point to each other; and (4) In case a foreign key is also part of a composite key, it points to the composite key (including itself).

Definition: Relational Graph. Given a relational model, pointers can be assigned using the pointer assignment rules. We then have a relational graph $G^L(R, A, T)$, where R is the set of relations, A is the set of attributes, and T is the set of pointers.

Definition: Minimal Attribute Granule. An attribute is a minimal granule if it cannot be partitioned into two attributes.

Definition: Attribute Correspondence. An attribute A corresponds to another attribute B in a different database if they are semantically similar and contain the same minimal attribute granule(s).

Definition: Type of Attribute Correspondence. There are four types of attribute correspondence between two attributes, A and B: *no match*, *equivalence, overlap,* and *inclusion*. Equivalence correspondence occurs where A and B have the same granule(s). Overlap is when both contain additional granule(s) besides the common granule(s). Inclusion correspondence occurs if A engulfs B, or vice versa. Otherwise, there is no match between A and B.

Definition: Attribute Correspondence Matrix. ACM = $\{A_{ij}, i[0,1,...,m], j[0,1,...,n]\}$, where A_{00} = "Federated Attribute", $\{A_{01}, A_{02}, ..., A_{0n}\}$ = names of the databases, $\{A_{10}, A_{20}, ..., A_{m0}\}$ = the federated attributes, $\{A_{ij}, i[1,...,m], j[1,...,n]\}$ = names of the local attributes, n is the number of component databases, and m is the number of federated attributes. For any $i[1,...,m]$, A_{ij} correspond to one another for all $j[1,...,n]$, and each federated attribute A_{i0} is a minimal attribute granule.

Definition: Object Correspondence. Given a set of ${}^{S}{}_{i}({}^{S}{}_{i}, {}^{S}{}_{i}, {}^{S}{}_{i}, {}^{S}{}_{i})$, $i\{1, 2, ..., n\}$, object correspondence can be derived based on the semantic meaning of objects in all component databases.

Definition: Type of Object Correspondence. There are four types of object correspondence between two objects, A and B: *no match, equivalence, overlap,* and *inclusion*. Equivalence correspondence occurs where objects A and B have the same attribute granules derived from their attributes. Overlap is when each has additional attribute granules besides the common granule(s). Inclusion correspondence occurs if A engulfs B, or vice versa. Otherwise, there is no match between A and B.

Definition: Canonical Object. An objects is referred to as a canonical object if all its attributes are minimal attribute granules.

Definition: Canonical Object Set. A set of objects is referred to as a canonical object set if its objects are all canonical.

Definition: Object Correspondence Matrix (OCM). We refer to the matrix representing the correspondences among objects as object correspondence matrix. OCM = $\{O_{ij}, i[0,1,...,l], j[0,1,...,n]\}$, where O_{00} = "Federated Object", $\{O_{01}, O_{02}, ..., O_{0n}\}$ = names of the databases, $\{O_{10}, O_{20}, ..., O_{l0}\}$ = names of the

federated objects, $\{O_{ij}, i[1,...,l], j[1,...,n]\}$ = names of the local objects, n is the number of databases, and l is the number of federated objects.

Algorithm: Derivation of MQG. Given a set of $G^{S}_{i}(^{S}_{i}, ^{S}_{i}, ^{S}_{i}, ^{S}_{i})$, $i\{1, 2, ..., n\}$, derive OCM and ACM. Let ^{M} = $\{O_{i0}, i[1,...,l]\}$, $^{M} = {}^{S}_{i}{}^{S}_{i}$, $^{M}={}^{S}_{i}{}^{S}_{i}$, $^{M} = \{A_{i0}, i[1,...,m]\}$, and ^{M} = all database names. Note that the union operations in ${}^{S}_{i}{}^{S}_{i}$ and ${}^{S}_{i}{}^{S}_{i}$ require transformation of the objects and attributes according to the object correspondences in OCM and the object correspondences in ACM. Therefore, we have $^{M}(^{M}, ^{M}, ^{M}, ^{M}, ^{M})$.

Definition: Canonical MQG. If the objects of MQG is a canonical object set and all its object hierarchies are explicitly defined, it is referred to as a canonical MQG.

Theorem: Using canonical MQG, adding a new database to the federation requires only append-only operations to the MQG and ACM, and consequently, canonical MQG results in high extensibility.

Proof: Adding a new database requires mapping $^{M}(^{M}, ^{M}, ^{M}, ^{M}, ^{M})$ to $^{S}_{i}(^{S}_{i}, ^{S}_{i}, ^{S}_{i}, ^{S}_{i})$, $i = n+1$. If ^{M}, ^{M}, and ^{M} map to $^{S}_{i}$, $^{S}_{i}$, and $^{S}_{i}$, respectively, and ^{M} subsumes $^{S}_{i}$, then inset the database name into ^{M} and add a column to the ACM. Otherwise, new objects, object hierarchies, attributes, and paths need to be added to ^{M}, ^{M}, ^{M}, and ^{M}, respectively. Modification to the elements in the MQG is not needed due to the canonical property.

4. Query Formulation using the MQG

Definition: Dynamic MQG. In dynamic MQG, objects in ^{M}, object hierarchies in ^{M}, and attributes in ^{M}, and database names in ^{M} of the MQG are displayed, but not paths in ^{M}. The user can point and click on database names and attribute (objects) names, and only paths relevant to the selected attributes or objects will be displayed for the user to choose. The user can also define selection constraints using pop-up boxes. The results of user interaction will then be converted to SQLs to the chosen databases.

Definition: Ambiguous Query. This concept is first defined in the context of universal relations (UR) and means that a query defined in a SQL under the UR model can be interpreted in different ways due to multiple access paths between two relations [4]. Ambiguous queries is one of the reasons that prevented UR model from wide spread adoption

Theorem: Query Disambiguation. The dynamic MQG eliminates ambiguous queries using semantic associations.

Proof: Given a dynamic MQG, $^{M}(^{M}, ^{M}, ^{M}, ^{M}, ^{M})$, the user expresses a federated query by selecting objects from ^{M}, attributes from ^{M}, and semantic associations from ^{M}, and databases from ^{M}. Since the associations identify the access paths in SQGs, the user query cannot have ambiguous queries.

5. An Example

Next, we give an example in order to give some idea on how the theory presented above can help with solving the problem of effective and user-friendly access of multiple and heterogeneous databases.

Shown in Figures 1 to 3 are the relational graphs for databases DB1, DB2, and DB3, which demonstrate the existence of various heterogeneities such as structure, abstraction, and naming heterogeneity. Figure 4 is the relational graph for the federated database that integrate the three databases. Figure 5 is the MQG for an example query.

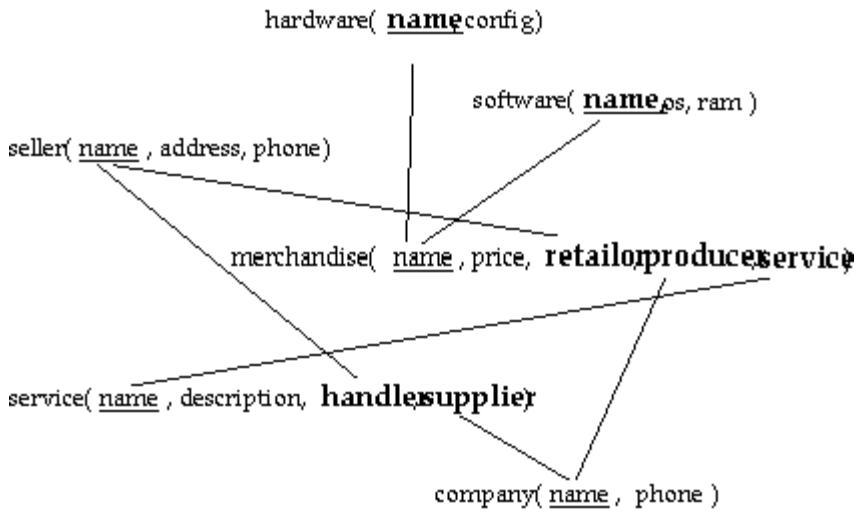

Figure 1. Relational Graph for Database DB1

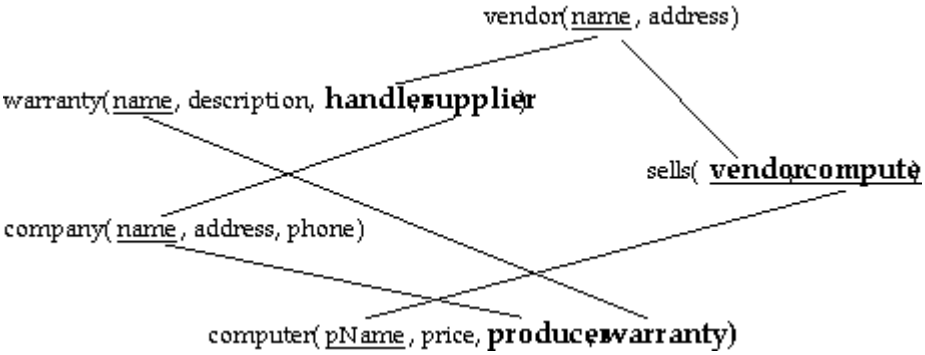

Figure 2. Relational Graph for Database DB2

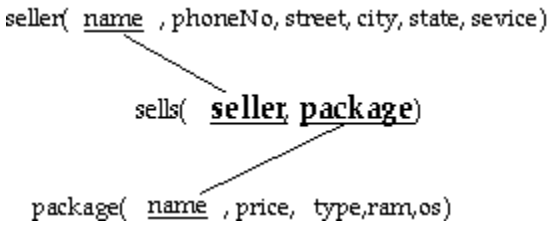

Figure 3. Relational Graph for Database DB3

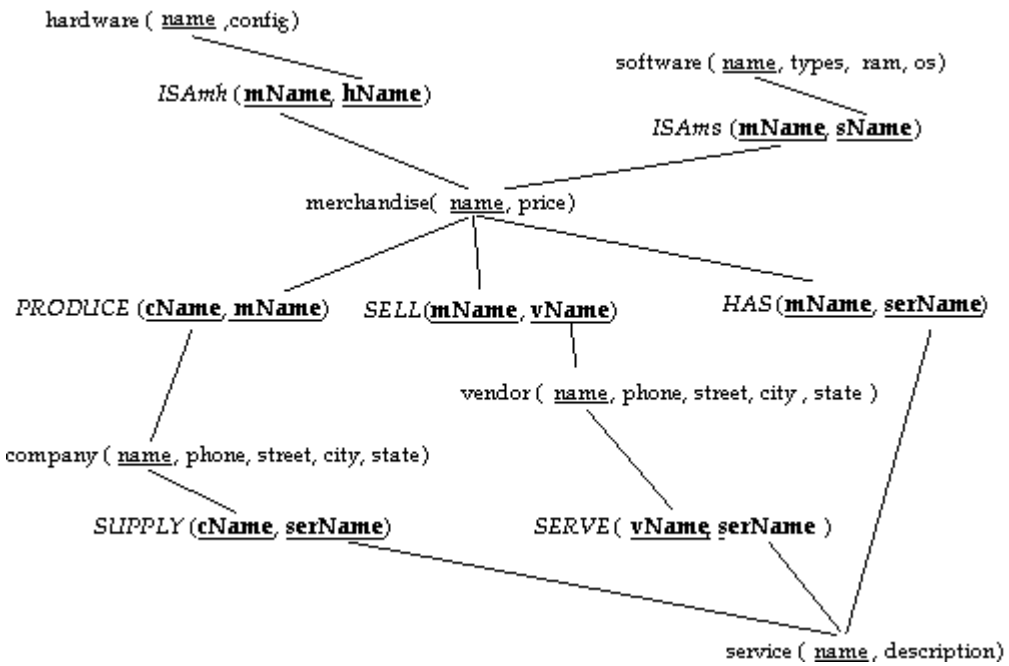

Figure 4. The Relational Graph for the Federated Database

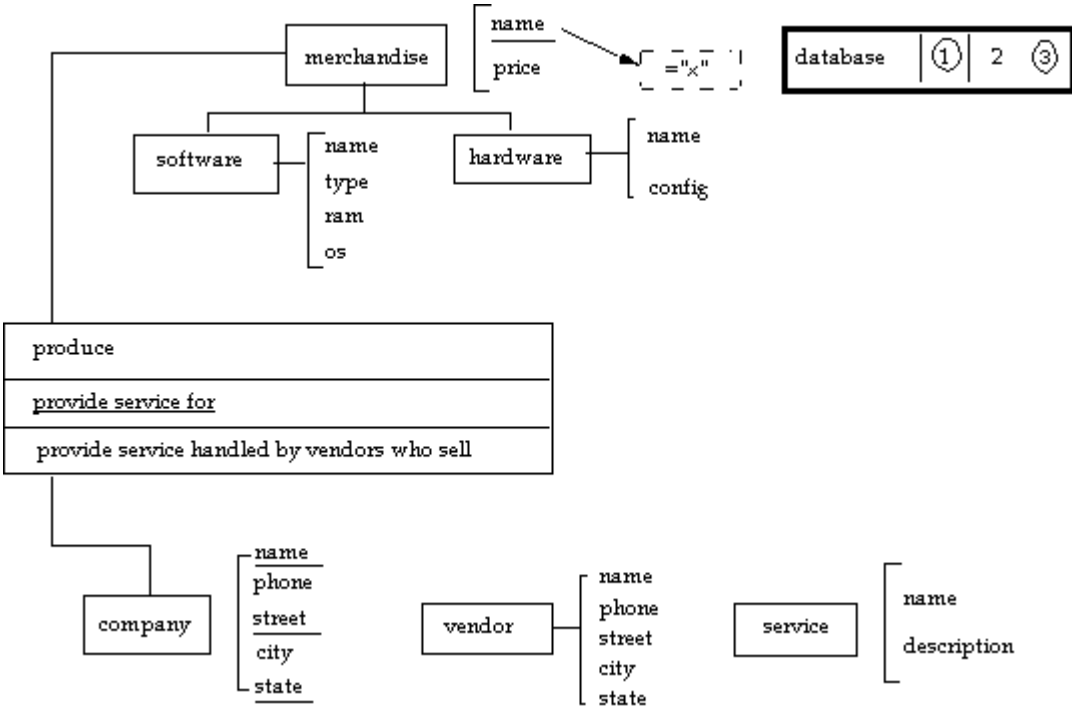

Figure 5. The MQG for a Query Example

The query example is "to find all information of companies that provide service for merchandise *x* from DB1 and DB3".

Figure 5 shows that there are three associations between object merchandise and object company. The association "provide service for" has been selected by the user. The MQG also allows the user to select DB1 and DB3, assign the constraint merchandise.name = "x", and select all attributes on company. The MQG has been constructed based on the theoretical framework highlighted in the paper. The MQG framework enables the capture of user query intention in a user-friendly manner and the translation of the federated query to component queries. However, due to space limitation, the details are omitted and can be found in [14].

6. Summary

In this short paper, we outlined the theoretical foundation for the multidatabase query graph (MQG) technique, which improves on the schema coordination approach in two ways. First, it makes the query interface more user-friendly by means of a dynamic MQG. The user can interact with the MQG to formulate queries to multiple databases. Second, the MQG technique helps eliminate query ambiguities that have troubled the universal relation model [4,7].

The MQG technique can help resolve naming, abstraction, and structural differences [5,8] but does not resolve differences in domain and integrity constraints [6,9]. The MQG technique maps the objects, attributes, and associations at the federated level with those at the component database level. During the process of mapping attributes and objects, naming and abstraction differences between databases are identified and resolved by means of a set of minimal attribute granules and a set of canonical objects. The structural differences between databases are recorded but not resolved. However, the MQG technique does not resolve differences in domain, and integrity constraints, but nearly informs the user when necessary.

References

1. Azarbod, C. and W. Perrizo. *CIKM* 1993.
2. Batini, M. et al. ACM Comp. Surveys, 18(4) 1986.
3. Burgess, C. G. *J. on Man-Machine Studies*, 1991(34).
4. Chang, Tzy-Hey and Edward Sciore. *IEEE TKDE*, 4(1), 1992.
5. Kim, W. and J. Seo. IEEE Computer, 24(12) 1991.
6. Larson, J. A. et al. IEEE TSE, 15(4) 1989.
7. Maier, D. et al. 12(3):317--335, 1987.
8. Naiman, C.E. and Ouksel, A.M. J. Org. Comp. 5(2) 1995.
9. Ramesh, V. and S. Ram. HICSS 1995.
10. Reddy, M. P. et al. IEEE TKDE 6(6) 1994.
11. Troxel, M., *8th Symp. on Meth. for Intelligent Sys.* 1994.
12. Zhao, J. L. DSS (forthcoming).
13. Zhao, J. L., A. Segev, and A. Chatterjee. ICDE95
14. Zhao and Yuan, Working Paper.